UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/902,818	07/10/2001	Paul F. Ringseth	MSFT-0670/180589.1	9208

41505 7590 04/25/2005

WOODCOCK WASHBURN LLP
ONE LIBERTY PLACE - 46TH FLOOR
PHILADELPHIA, PA 19103

EXAMINER	
CHOW, CHIH CHING	
ART UNIT	**PAPER NUMBER**
2192	

DATE MAILED: 04/25/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/902,818	RINGSETH ET AL.	
	Examiner	Art Unit	
	Chih-Ching Chow	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *27 December 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-42* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-22, 33-42* is/are rejected.

7)☒ Claim(s) *23-32* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *10 July 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to amendment dated December 27, 2004.

2. Per Applicants' request, specification and claims 2, 27, 29 have been amended. Claims 1-42 remain pending.

Response to Amendment

3. Applicants' amendment dated 12/27/2004, responding to the 09/27/2004 Office action provided in the objection of claims. The examiner has reviewed the updated claims, the objection to the claims 2, 27 and 29 are hereby withdrawn in view of Applicants' amendment.

4. Applicants' amendment dated 12/27/2004, responding to the 09/27/2004 Office action provided in the rejection of claims 1, 8 and 11 for the 'provider object'. The examiner has reviewed the updated specification, and noted that new matter has been introduced into the disclosure. See Objection to the Specification herein below.

Response to Arguments

5. Applicant's arguments for Claims 1-42 have been fully considered respectfully by the examiner but they are not persuasive. See MPEP 7.38 Arguments are Moot because of new ground(s) of rejection Applicant's arguments with respect to claims 1, 8, and 11 have been considered but are moot in view of the new ground(s) of rejection. The amended parts, "Also, the term 'provider object' refers to an object that is a provider. A provider may query the compiler for state or information from which the provider may conditionally inject source code or

more abstractly manipulate the parse tree", were never mentioned in the specification before.

Specification

6. The amendment filed 12/27/2004 is objected to under 35 U.S.C. 132(a) because it introduces new matter into the disclosure. 35 U.S.C. 132(a) states that no amendment shall introduce new matter into the disclosure of the invention. The added material which is not supported by the original disclosure is as follows: "query the compiler for state of information" and "the provider may conditionally inject new source code or more abstractly manipulate the parse tree" were never mentioned in the specification before.

Applicant is required to cancel the new matter in the reply to this Office Action.

Claim Rejections - 35 USC § 102

7. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless -
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

8. Claims 1-6, 8-11, 14-22, 33-42 rejected under 35 U.S.C. 102(e) as being anticipated by US2003/0023957, by Bau et al. (hereinafter, "Bau").

CLAIM	Bau
1. A method for implementing (Simple Object Access Protocol) SOAP-based Web services via a programming language in a computing system, comprising:	Bau teaches a method of using SOAP-based Web services via a programming language in a computing system, see Bau paragraph 7, "for example, in order to generate and deploy the most **basic of web services**, **developers** are currently required to implement one or more mechanisms for: **Sending and receiving data** via internet protocols; parsing and generating message bodies and headers packaged using protocols such as the **Simple Object Access Protocol (SOAP)**" - SOAP-based Web services, and paragraph 23, "In one embodiment of the present invention, the developer expresses the logic offered by the **web service** using a standard **programming language** augmented with **declarative annotations specifying preferences for exposing that logic as a web service**". For item (1), see Bau's FIGs 3A, 3B, 3C, and 4 all with declaration of SOAP handling mechanism. For items (ii) and (iii), see Bau paragraph 68, "If an annotation is found, compiler 506 determines (*compiling the said code*) the annotation type (e.g. conversational, asynchronous, and so forth), block 608, identifies the statement or declaration it modifies, block 609, identifies and **generates a set** of **helper objects** *(generates one of additional code)* to be hooked up to the object file being compiled at **runtime**, block 610, and associates meta-data extracted from the
(i) in connection with code that implements at least one SOAP-based Web service, declaring at least one SOAP handling mechanism corresponding to at least one SOAP-based Web service via a construct of said programming language;	
(ii) when compiling said code, communicating with a provider object; and	
(iii) generating at least one of additional code and data for use at run-time when at least one of sending and receiving a SOAP message for said at least one SOAP-based Web service occurs.	

annotations with the object file, block 611." And paragraph 77, "once invoked, any web service method can use the proxy objects created for **generating calls** to the client or external services. The proxy objects will map objects passed as parameters into corresponding XML elements and use them and the remote method name to generate and **send** a message (*sending a SOAP message*) to the remote service or client. In the case of external services, the message is generated and sent using the protocol specifics (e.g., **SOAP** over HTTP) specified by the associated service description file. In the case of the client, the message is generated and sent using the protocol specifics used by the client in the initial start method. In one embodiment, synchronous responses from the client or remote services are parsed to extract the result and map it onto a **representative object** *(provider object)*, which is returned to the web service code (*receiving a SOAP message*) as the **return value** of the proxy object method invocation."

2. A method according to claim 1, wherein said declaring includes declaring at least one SOAP handling mechanism corresponding to at least one SOAP-based Web service via a C++ construct.

For the features of claim 1 see claim 1 rejection. Bau does not specify a certain programming language, as long as it's a 'standard programming language', see Bau paragraph 23, "the developer expresses the logic offered by the web service using a **standard programming language** augmented with declarative annotations

specifying preferences for exposing that logic as a web service." - C++ is consider as a 'standard programming language', and using a 'C++ construct' is consider as an implementation detail.

3. A method according to claim 2, wherein said declaring includes declaring at least one C++ attribute corresponding to at least one SOAP-based Web service.

For the features of claim 1 see claim 1 rejection. Bau's disclosure also provides attribute corresponding to at SOAP-based Web service, see Bau's paragraph 32, "using IDE 111 a developer is able to identify which methods (if any) should be buffered, which methods should be asynchronous, which methods should be stateful, and which methods are to cause a non-isomorphic mapping between incoming message elements and native language objects for example. Furthermore, once a method has been added to the web service (e.g. via graphical manipulation by the developer), IDE 111 provides the developer with the ability to further define and/or modify the method by specifying one or more parameters and/or **attributes**." And see Bau, 25, "Enhanced web services 104 includes annotated source code 107, enhanced compiler 106, and various deployed service components 108. As will be discussed in further detail below, when annotated source code 102 is compiled by enhanced compiler 106, the **compiler generates one or more object files, software components and deployment descriptors to facilitate the automated deployment of web**

	service components 108."
4. A method according to claim 1, wherein when sending a SOAP message for said at least one SOAP-based Web service, the method further includes utilizing said at least one of additional code and data at run-time to generate the SOAP message.	See claim 1 rejection.
5. A method according to claim 1, wherein when receiving a SOAP message for said at least one SOAP-based Web service, the method further includes utilizing said at least one of additional code and data at rum-time to convert said SOAP message to an object of the programming language.	Same as claim 1 rejection (*helper object*).
6. A method according to claim 1, wherein said declaring reduces the amount of code a developer writes relative to writing the code without said declaring.	For the features of claim 1 see claim 1 rejection. Bau's declaring in FIGs. 3A-C, and 4, declared it's running on a SOAP-based protocol, it uses existing SOAP-based message structure (see Envelop, header, Body, etc.) it sure reduces the amount of code than without a SOAP declaration.
8. A method according to claim 1, wherein said provider object communicates with the compiler of said code to generate said at least one of additional code and data.	For the features of claim 1 see claim 1 rejection. For the rest of claim 8 feature see FIG. 6 and claim 1 rejction.

9. A method according to claim 1, wherein via said declaring, said method hides from the developer the underlying details regarding the SOAP protocol, dispatching to the appropriate object and function, marshaling the (extensible Markup Language) XML, un-marshaling the XML, and generating the SOAP response.	For the feature of claim 1 see claim 1 rejection. For the rest of claim 9 feature see paragraph 7, "in order to generate and deploy the most basic of web services, developers are currently required to implement one or more mechanisms for: Sending and receiving data via internet protocols; parsing and generating message bodies and headers packaged using protocols such as the **Simple Object Access Protocol (SOAP);** controlling access to services in a secure way; mapping data between **XML messages** and internal data structures within the web service logic; transacting operations so they are reliable and predictable;"
10. A method according to claim 1, wherein the underling XML packaging of the SOAP message is transported according to said at least one of sending and receiving via at least one of hypertext transfer protocol (HTTP), file transfer protocol (FTP), transmission control protocol (TCP), user datagram protocol (UDP), internet relay chat (IRC), telnet protocol and Gopher protocol.	For the features of claim 1 see claim 1 rejection. See Bau's paragraph 24, "networking fabric 100 represents one or more interconnected data networks, such a the Internet or World Wide Web, that are equipped to employ one or more well-known communication protocols such as the **hypertext transfer protocol (HTTP)**" and paragraph 35, "IDE 111 may communicate with web server 102 to create such files and directories via networking fabric 100 using one or more network protocols such as the **file transfer protocol (FTP)**".
11. A method according to claim 1, wherein said provider object is an attribute provider object and said	For the features of claim 1 see claim 1 rejection. See Bau claim 16, "generating one or more **object codes** (*provider*

attribute provider object has access to attribute type and marshaling information.	*object*) defining one or more **publicly accessible service components** based at least in part upon the source code; generating meta-data based at least in part upon the one or more declarative annotations; associating the meta-data with the one or more object codes."
14. A method according to claim 1, wherein said declaring includes declaring at least one of a soap-handler attribute, a soap-method attribute and a soap-header attribute.	For the features of claim 1 see claim 1 rejection. For the rest of claim 14 feature, see Bau's FIG. 3B, A 'Soap: Header' is declared.
15. A method according to claim 14, wherein said at least one of a soap-handler attribute, a soap-method attribute and a soap-header attribute are declared in the at least one Web service's class declaration.	Same as claim 14 rejection.
16. A computer readable medium bearing computer executable instructions for carrying out the method of claim 1.	Bau's disclosure is also a computer readable medium to carry out the claim 1 method listed above.
17. A modulated data signal carrying computer executable instructions for performing the method of claim 1.	Same as claim 1 rejection; Bau's method also uses a modulated data signal carrying computer executable instructions.
18. A computing device comprising means for performing the method of claim 1.	Same as claim 1 rejection; Bau's method also uses a computing device to perform the features recited in claim 1.

19. A method of compiling programming language code with a compiler, comprising: (i) initially parsing the code; (ii) during said initial parsing, encountering an attribute block and allowing an interface definition that follows the attribute block to include embedded information ; and (iii) parsing the interface definition.	See Bau's paragraph 65, "Once compiler 506 receives annotated source code files 502, parser 505 reads and **parses** programming statements contained within the source code. In one embodiment, compiler 506 is enhanced to recognize annotations based on an extended syntax for specifying functionality of the source file to be deployed as a web service. Accordingly, **as parser** 505 **parses** the annotated source code, it identifies the presence and composition of embedded annotations based on this extended syntax. In one embodiment of the present invention, **compiler** 506 infers by way of the source code annotations (*code annotations can be attribute block*) the **interface of the web service** (*compiling the interface definition*) that is to be exposed to remote clients, the interface of services that will be called by the runtime to be created by compiler 506, as well as internal storage requirements and persistence behavior of the web service."
21. A method according to claim 19, further including determining whether said attribute block includes an object attribute.	For the features of claim 19 see claim 19 rejection. For the rest of claim 21 feature, see claim 1 rejection.
22. A method according to claim 21, wherein for each function in the interface definition, the compiler parses the function declaration, and	For the features of claim 21 see claim 21 rejection. For the rest of claim 22 feature, see claim 1 rejection.

saves the parameter attribute information for use later in determining a (Simple Object Access Protocol) SOAP message structure.

33. A computer readable medium bearing computer executable instructions for carrying out the method of claim 19.

Bau's disclosure is a computer readable medium that carrying out the fetuares recited in claim 19.

34. A modulated data signal carrying computer executable instructions for performing the method of claim 19.

Same as claim 19 rejection; Bau's method also uses a modulated data signal carrying computer executable instructions.

35. A computing device comprising means for performing the method of claim 19.

Same as claim 19 rejection; Bau's method also uses a computing device to perform the features recited in claim 19.

36. A method processing (Simple Object Access Protocol) SOAP messages at run-time in a computing system having a server with at least one application having at least one user object implementing at least one SOAP-based Web service, comprising:

at least one of receiving an incoming SOAP message from a client computing device and receiving a call from a user object with at least one programming language construct; and

with a runtime intermediate layer, at least one of converting said

Same as claim 1 rejection. Where see Bau paragraph 68, "If an annotation is found, compiler 506 determines (*compiling the said code*) the annotation type (e.g. conversational, asynchronous, and so forth), block 608, identifies the statement or declaration it modifies, block 609, identifies and **generates a set** of **helper objects** *(generates one of additional code)* to be hooked up to the object file being compiled at **runtime**, block 610 (*intermediate layer*), and associates meta-data extracted from the annotations with the object file, block 611."

incoming SOAP message to a corresponding programming language construct and converting said call with at least one programming language construct to an outgoing SOAP message, wherein said runtime intermediate layer utilizes at least one of code and data generated during compilation of said user object.	
37. A method according to claim 36, wherein said programming language is C++.	For the features of claim 36 see claim 36 rejection. Bau does not specify a certain programming language, as long as it's a 'standard programming language', see Bau paragraph 23, "the developer expresses the logic offered by the web service using a **standard programming language** augmented with declarative annotations specifying preferences for exposing that logic as a web service." - C++ is consider as a 'standard programming language'
38. A method according to claim 36, wherein said converting of said incoming SOAP message to a corresponding programming language construct includes determining the appropriate user object to receive the SOAP message.	For the features of claim 36 see claim 36 rejection. For the rest of claim 38 rejection see claim 1 rejection.
39. A method according to claim 36, wherein said converting said call with at least one programming language construct to an outgoing SOAP message includes formatting	For the features of claim 36 see claim 36 rejection. For the rest of the features of claim 39 see claim 1 (iii) rejection.

the message for delivery to an intended client computing device.	
40. A computer readable medium bearing computer executable instructions for carrying out the method of claim 36.	Bau's disclosure is a computer readable medium that carrying out the fetuares recited in claim 36.
41. A modulated data signal carrying computer executable instructions for performing the method of claim 36.	Same as claim 36 rejection; Bau's method also uses a modulated data signal carrying computer executable instructions.
42. A computing device comprising means for performing the method of claim 36.	Same as claim 36 rejection; Bau's method also uses a computing device to perform the features recited in claim 36.

Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 7, 12, 13 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bau et al, US2003/0023957 (hereinafter "Bau"), further in view of "Mapping CORBA and SOAP", 10/31/2000 (hereinafter "CORBA").

CLAIM	Bau / CORBA
7. A method according to claim 1, wherein said declaring includes describing at least one Web service interface using embedded interface definition language (IDL).	For the features of claim 1 see claim 1 rejection. Bau teaches "interface definition" in paragraph 28, "In one embodiment, web server 102 communicates with enterprise server 115 and/or 120 based upon their **public interfaces advertised by WSDL files** 117 and/or 127." Bau does not mention 'IDL' specifically. However, CORBA shows the feature in an analogous art. On CORBA, page 7, under XORBA example has shown using IDL on a SOAP-based Web service interface. It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement of the SOAP-based Web service with the Interface Definition Language (IDL) further taught by CORBA for the purpose of transmitting objects via a SOAP-based protocol via the internet (see CORBA page 6, 2nd paragraph).
12. A method according to claim 1, wherein for each method of said code that is to be exposed as part of the at least one Web service, the method is introduced via an interface as part of an interface definition language (IDL) description.	For the features of claim 1 see claim 1 rejection. For the rest of claim 12 feature see claim 7 rejection.
13. A method according to claim 12, wherein said declaring further includes specifying at least one of in,	For the features of claim 12 see claim 12 rejection. For the rest of the features of claim 13 see claim 7 rejection. The

out, size-is and retval IDL attributes in the at least one Web service's interface declaration.	XORBA example on CORBA page 7 has 'in' parameters.
20. A method according to claim 19, wherein said embedded information is implemented via interface definition language (IDL).	For the features of claim 19 see claim 19 rejection. For the rest of the features of claim 20 see claim 7 rejection.

Allowable Subject Matter

11. Claims 23-32 are objected to as being dependent upon a rejected base claim. The limitation recited in claims 23-32 when taken individually is not deemed allowable. However, claims 23-32 would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Conclusion

12. The following summarizes the status of the claims:

35 USC § 102 rejection : Claims 1-6, 8-11, 14-22, 33-42

35 USC § 103 rejection : Claims 7, 12, 13 20.

13. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Brodsky et al., US2002/0046294, teaches a method and a system for processing an enterprise application request crossing network, based on SOAP complint XML documents.

14. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Chih-Ching Chow whose telephone number is 571-272-3693. The examiner can normally be reached on 7:00am - 3:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306. Any inquiry of a general nature of relating to the status of this application should be directed to the **TC2100 Group receptionist: 571-272-2100**.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Chih-Ching Chow
Examiner
Art Unit 2192
April 15, 2005

C.C.

ANTONY NGUYEN-BA
PRIMARY EXAMINER